

## 1 A Further analysis

### 2 A.1 Other datasets

3 To estimate the generalization ability of FFDesign, we extend FFDesign on various real-world  
4 datasets, TS50, TS500 [1], and Ollikainen [2] datasets. The first two datasets include randomly  
5 selected 50 and 500 non-redundant proteins from PISCES server [3], respectively. Ollikainen dataset  
6 has 40 proteins sampled from diverse protein domains to explore structural constraints on amino  
7 acid covariation. For a download issue, 37 proteins are used in this work. We evaluate CATH4.2  
8 pre-trained PiFold and DNDesign-PiFold in a zero-shot setting. All results are shown in Table 1. In  
9 the case of a TS50 dataset with a small number of 50 proteins, DNDesign performs worse. However,  
10 when handling a larger dataset of 500 points in TS500, a gain in perplexity is observed. The Ollikainen  
11 dataset demonstrates performance gains in both perplexity and sequence recovery.

Table 1: Protein sequence design comparison on TS50, TS500, and Ollikainen. PP and SR indicate perplexity and sequence recovery, respectively.

Model	TS50		TS500		Ollikainen	
	PP	SR	PP	SR	PP	SR
PiFold	3.71±0.08	57.37±1.32	3.38±0.03	59.74±0.11	3.70±0.10	56.95±0.44
DNDesign-PiFold	3.77±0.02	55.58±0.37	3.34±0.01	59.55±0.45	3.68±0.06	57.42±0.32

### 12 A.2 Multi-chain sequence design

13 Designing a specific part of a protein assembly, which is a complex of single proteins, is one  
14 of the most common and important problems in protein design, such as the problem of fixing  
15 complementarity determining region (CDR) loop in an antigen-antibody complex [4]. To evaluate  
16 the performance of the inverse-folding models for these scenarios, we design the task of designing  
17 subunit protein sequences of protein assemblies. Following ProteinMPNN [5], we train models  
18 using protein assemblies from PDB [6]. In training, models are tasked to generate amino acids of a  
19 target chain when given a whole protein complex. For evaluation, we select 178 protein assemblies  
20 containing more than four protein chains from the test set, and conduct sequence design on multiple  
21 chains in alphabetical chain order. As shown in Table 2, DNDesign gives marginal gains.

Table 2: Comparison of FFDesign with previous models on multimer sequence design of protein heteromer assemblies. All results are newly calculated.

Model	1	2	3	4	Avg
PiFold	46.75	46.75	46.75	46.75	46.75
DNDesign-PiFold	46.70	46.90	46.94	46.74	46.82

### 22 A.3 Inference speed

23 We compared the inference time on the same machine (single V100) for all proteins in CATH4.3 test  
24 set. As shown in Table 3, there is no significant inference time difference compared to the previous  
25 methods. While DNDesign-PiFold performs better than ESM-IF (53.75% vs 51.60%) and have  
26 several times less parameters (5M vs 140M), DNDesign-PiFold has twice the faster inference rate.

Table 3: Inference speed on CATH 4.3 test set.

Model Type	Time(min)
ProteinMPNN	19
PiFold	20
DNDesign-PiFold	22
ESM-IF	67

27 **A.4 Ablation study**

28 In order to understand the effectiveness of each proposed operation, we conduct an ablation study as  
 29 shown in Table 4. All models are trained in the same setting for CATH 4.2 benchmark. Without  
 30 our proposed operations, the baseline model shows 4.70 and 48.06% of perplexity and recovery,  
 31 respectively. The best effective module is ForceEdge, which boosts improvements of -0.12 and 1.27%  
 32 over the baseline. Also, ForceInit and ForceNode modules benefit models on both perplexity and  
 33 recovery by -0.08, -0.06, and 0.46%, 0.70%, respectively. Other operations also show performance  
 34 gains over baselines. All results confirm that our proposed operations effectively transfer learned  
 35 folding physics knowledge to the entire networks.

Table 4: Ablation of five components of DNDesign.

	Perplexity ↓	Recovery % ↑
Baseline	4.70	48.06
ForceInit	4.62 (-0.08)	48.52 (+0.46)
ForceNode	4.64 (-0.06)	48.76 (+0.70)
ForceEdge	4.58 (-0.12)	49.33 (+1.27)
GlobalForce	4.65 (-0.05)	48.51 (+0.45)
ForceLogits	4.64 (-0.06)	48.76 (+0.70)
ForceInit + ForceNode	4.52 (-0.18)	49.66 (+1.60)
ForceInit + ForceNode + ForceEdge	4.52 (-0.18)	49.52 (+1.46)
ForceInit + ForceNode + ForceEdge + GlobalForce	4.52 (-0.18)	50.00 (+1.94)
ForceInit + ForceNode + ForceEdge + GlobalForce + ForceLogits	<b>4.46</b> (-0.24)	<b>50.00</b> (+1.94)

Table 5: Comparison of sequence recovery on CATH 4.2 with regard to structural environment: core and surface. We categorize residues by density of neighboring  $C_{\alpha}$  atoms within 10 angstroms of the central residue  $C_{\alpha}$  atom. Core and surface residues have more than 24 and less than 16 neighbors, respectively.

Model	Recovery % ↑	
	Core	Surface
PiFold	66.67	40.00
DNDesign-PiFold	<b>66.67</b>	<b>40.74</b>

Table 6: Comparison of sequence recovery on CATH 4.3 with regard to structural environment: core and surface.

Model	Recovery % ↑	
	Core	Surface
ESM-IF	39.00	72.00
DNDesign-PiFold (scaled-up)	<b>43.48</b>	<b>75.00</b>

36 **Recovery depending on the structural context** Table 5 shows the sequence recovery performance  
 37 of PiFold and DNDesign-PiFold on the core and surface residues of test proteins in the CATH4.2.  
 38 We observe that both models perform better in core residue design than the surface. The results are  
 39 in line with [7]. Also, we see that DNDesign-PiFold benefits on surface scenario. This is meaningful in  
 40 terms of applications because many protein applications, such as enzymes and antibody-antigens,  
 41 occur on surface residues rather than core residues. Also, Table 6 reveal that DNDesign-PiFold  
 42 effectively generate sequences in core and surface compared to ESM-IF, which has several times  
 43 larger parameters compared to DNDesign-PiFold.

44 **Force logit ratio ablation study** In order to confirm the inverse folding performance according to  
 45 the force logits during training and inference, we train DNDesign-PiFold on CATH4.2 by varying  
 46 the ratio (0.2, 0.4, 0.6, 0.8) during training. As shown in Table 7, all models have same sequence  
 47 recovery performance, meaning that mixing force logits to inverse-folding logits benefits with any  
 48 ratio. Also, the model with 0.2 force ratio shows the best performance. In addition to this, we ablate  
 49 the force ratio during inference using a model trained with 0.2 force logits ratio during training. As  
 50 shown in Table 8, 0.2 ratio gives best performance. With the empirical results, we use force logit  
 51 ratio of 0.2 for both training and inference.

Table 7: CHyper-parameter search of force logit ratio during training.

Ratio	0.2	0.4	0.6	0.8
Perplexity	4.46	4.47	4.51	4.50
Sequence recovery	50.0	50.0	50.0	50.0

Table 8: Hyper-parameter search of force logit ratio during inference.

Ratio	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Perplexity	4.58	4.48	4.46	4.56	4.84	5.38	6.32	7.87	10.33

## 52 B Others

### 53 B.1 Limitations

54 Inverse-folding, a powerful method that enables fine-grained protein engineering when a known  
55 structure exists, has been used in various protein engineering. Therefore, it has a limitation that  
56 it can be performed only when a known structure exists. Fortunately, the limitation is somewhat  
57 alleviated because models like AlphaFold2 [8], RoseTTAFold [9], and ESMFold [10] correctly  
58 predict structures of proteins that have not yet been experimentally characterized. However, there are  
59 proteins that these models fail, such as orphan proteins and antibody-antigen complexes, because  
60 the proteins lacks multiple sequence alignments that is key inputs for those models. Additionally,  
61 the requirement that the structures in the dataset need to be locally optimized (with low energy) is  
62 the bottleneck for FFDesign training. We think that numerous samples in the predicted structures do  
63 not meet the requirement. So, we suspect that these structures may negatively impact the learning of  
64 the local minimized structure distribution if included in the training dataset. However, we believe  
65 that the two constraints brought by the aforementioned structure dependency can be resolved with  
66 better structure prediction. Meanwhile, recently, Dauparas et al [5] has experimentally verified in  
67 web-lab that protein design is possible more successfully than previously widely used computational  
68 methods by combining an inverse-folding model with a structure generation model. In conclusion,  
69 even though inverse-folding has structural dependencies, the design method is still a powerful strategy  
70 for designing multiple proteins that match the target fixed structure.

### 71 B.2 Broader impact

72 Proteins are a promising material used in numerous fields, such as drug development, enzymes, and  
73 carbon capture. Therefore, designing proteins that meet specific objectives is crucial but challenging  
74 because even a single amino acid change can lead to significant property changes. However, one  
75 structure can have numerous sequence combinations of 20 amino acids. Optimizing and narrowing  
76 candidates ultimately allows us to save considerable human and material resources required for  
77 extensive experiments in candidate screening through wet-lab experiments, which leads to significant  
78 environmental and economic benefits. We expect that our approach is a promising solution to gain  
79 the mentioned advantages.

## 80 C Training details

81 We provide the main hyperparameters of DNDesign below.

## 82 D Node and Edge Featurization

83 **Distance features** Distances between atom pairs are essential for quantum mechanical inter-  
84 actions. We use euclidian distances between atom pairs. When computing inter-residue dis-  
85 tances for node features, we sample intra-residue atom pairs from  $A \in \{C_i, C_{\alpha i}, N_i, O_i\}$  and  
86  $B \in \{C_i, C_{\alpha i}, N_i, O_i\}$ . When computing inter-residue distances for edge features, we adopt pairs  
87 using from  $A \in \{C_i, C_{\alpha i}, N_i, O_i\}$  and  $B \in \{C_j, C_{\alpha j}, N_j, O_j\}$ . To enrich the distance information,

Table 9: DNDesign-PiFold hyperparameters.

	DNDesign-PiFold	DNDesign-PiFold (uniref)
Structure encoder layers	2	4
Sequence decoder layers	2	4
Folding physics learning layers	4	4
Transformer embedding dim	128	128
Attention heads	4	4
Top K neighbors	30	30
Batch size (tokens per GPU)	6000	6000
GPUs	1	1
CATH:AF2 mixing ratio	1:0	1:6
Epochs	100	150
Optimizer	Adam	Adam
$\beta_1$	0.9	0.9
$\beta_2$	0.999	0.999
Learning rate schedule	OneCycleLR	OneCycleLR
Learning rate	1e-3	1e-3
Annealing strategy	Cosine	Cosine
Increasing learningrate steps percent	30%	30%

88 we apply radial basis function on the distances as follows:

$$D_{j \rightarrow i, r} = \exp\left(-\frac{(\|x_j - x_i\|)^2}{\sigma^2}\right) \quad (1)$$

89 We use 16 basis so that there are each pair has 16 distance-related features. Unlike [11], we do not  
90 use virtual atoms.

91 **Angle features** We use the bond angles  $\alpha_i, \beta_i, \gamma_i$  and torsion angles  $\psi_i, \pi_i, \omega_i$  from  
92  $C_{i-1}, N_i, C_{\alpha i}, C_i$ , and  $N_{i+1}$  as node features between adjacent  $(i-1, i, i+1)$  residues. For angle-  
93 related edge features, we adopt the quaternions of relative rotation between their local coordinate  
94 systems as follows:

$$q_{ij} = q(g_i^T g_j) \quad (2)$$

95 where  $q$  is the quaternion encoding operation.

96 **Direction features** We use relative directional information between orientation  $Q_i$  of a residue  $i$   
97 and directions of intra-residue  $i$  or inter-residue  $j$  atoms to  $C_{\alpha i}$  as follows:

$$R_i = g_i^T \frac{a_i - C_{\alpha i}}{\|a_i - C_{\alpha i}\|}, R_{ji} = g_i^T \frac{a_j - C_{\alpha i}}{\|a_j - C_{\alpha i}\|} \quad (3)$$

98 We note that  $a$  includes  $C, N, C_{\alpha}, O$  and  $C, N, C_{\alpha}, O$  for node features and edge features, respec-  
99 tively.

## 100 References

- 101 [1] Zhixiu Li, Yuedong Yang, Eshel Faraggi, Jian Zhan, and Yaoqi Zhou. Direct prediction of pro-  
102 files of sequences compatible with a protein structure by neural networks with fragment-based  
103 local and energy-based nonlocal profiles. *Proteins: Structure, Function, and Bioinformatics*,  
104 82(10):2565–2573, 2014.
- 105 [2] Noah Ollikainen and Tanja Kortemme. Computational protein design quantifies structural  
106 constraints on amino acid covariation. *PLoS computational biology*, 9(11):e1003313, 2013.
- 107 [3] Guoli Wang and Roland L Dunbrack Jr. Pisces: a protein sequence culling server. *Bioinformatics*,  
108 19(12):1589–1591, 2003.
- 109 [4] Paul J Carter. Potent antibody therapeutics by design. *Nature reviews immunology*, 6(5):343–  
110 357, 2006.

- 111 [5] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F  
112 Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep  
113 learning-based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.
- 114 [6] Joel L Sussman, Dawei Lin, Jiansheng Jiang, Nancy O Manning, Jaime Prilusky, Otto Ritter, and  
115 Enrique E Abola. Protein data bank (pdb): database of three-dimensional structural information  
116 of biological macromolecules. *Acta Crystallographica Section D: Biological Crystallography*,  
117 54(6):1078–1084, 1998.
- 118 [7] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and  
119 Alexander Rives. Learning inverse folding from millions of predicted structures. In *International  
120 Conference on Machine Learning*, pages 8946–8970. PMLR, 2022.
- 121 [8] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ron-  
122 neberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al.  
123 Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.
- 124 [9] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov,  
125 Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate  
126 prediction of protein structures and interactions using a three-track neural network. *Science*,  
127 373(6557):871–876, 2021.
- 128 [10] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin,  
129 Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level  
130 protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.
- 131 [11] Zhangyang Gao, Cheng Tan, and Stan Z Li. Pifold: Toward effective and efficient protein  
132 inverse folding. *arXiv preprint arXiv:2209.12643*, 2022.